# Improving the State Space Organization of Untrained Recurrent Networks

Michal Čerňanský[1], Matej Makula[1], and Ľubica Beňušková[2]

[1] Faculty of Informatics and Information Technologies, STU Bratislava, Slovakia
[2] Department of Computer Science, University of Otago, Dunedin, New Zealand
{cernansky,makula}@fiit.stuba.sk, lubica@cs.otago.ac.nz

**Abstract.** Recurrent neural networks are frequently used in cognitive science community for modeling linguistic structures. More or less intensive training process is usually performed but several works showed that untrained recurrent networks initialized with small weights can be also successfully used for this type of tasks. In this work we demonstrate that the state space organization of untrained recurrent neural network can be significantly improved by choosing appropriate input representations. We experimentally support this notion on several linguistic time series.

## 1 Introduction

Many commonly used real-world data with time structure can be expressed as a sequence of symbols from finite alphabet like symbolic time series. Since their emergence, neural networks were applied to symbolic time series analysis. Especially popular is to use connectionist models for processing of complex language structures.

Recurrent neural networks (RNNs) are frequently applied to processing symbolic sequences. Elman's simple recurrent network (SRN) [1] is the most frequently used architecture. Common algorithms usually used for RNN training are based on gradient minimization of the output error. Several advanced approaches have been proposed, especially popular are methods based on the Kalman filtration.

But thorough training process may not be necessary. In [2] we have studied the dynamics of untrained recurrent network randomly initialized with small weights. We have shown that for some tasks the contractive dynamics of such a recurrent network is sufficient and the structural organization of the state space can be used to create prediction machines having comparable performance to thoroughly trained SRNs. We have shown and explained strong correspondence between such a prediction machines and a class of Markov models – variable length Markov models (VLMMs) [3]. On the other side carefully trained RNNs can usually achieve significantly better performance.

A lot of attention is now being focused on connectionist models known under the name "reservoir computing". The most prominent example of these approaches is a recurrent neural network architecture called an echo state network (ESN). ESNs were successfully applied in multiple sequence processing tasks [4] and recently several authors have also used ESNs for language modeling. In [5] ESN was used to the next symbol prediction task on the "Little Red Riding Hood" fairy tale. In [6] authors studied performance of ESN and SRN in modeling natural-like artificial languages. Authors

claim that ESN show similar predictive performance to SRN, superior to Markov models of various orders (n-grams). In [7, 8] authors also used ESNs to process natural-like languages but in the context of studying systematicity, that is defined as the ability to process novel sometimes even ungrammatical sentences.

In [9] we have studied the state space organization of the recurrent neural network before and after training on three artificial languages. We found that the dynamics of the trained network is still based on the fixed point attractors, but these attractors do not correspond to the symbols representing the words as is the case of the untrained network. Instead attractors correspond to the word categories enabling the network to better manage its state space. This behavior was achieved by the training process by setting input weights sourced from the inputs of the same category to the similar values.

Input representations, i.e. weights spreading information from active input neuron are usually randomly initialized in models based on the contractive dynamics such as ESNs. In this work we study and evaluate word co-occurrence method of initialization of input weights on two models based on random contractive dynamics: Neural prediction machines (NPMs) and echo state networks (ESNs). We compare their predictive performance on two linguistic datasets and elaborate upon importance of input representations on the state space organization in these models. Such an enhanced ESNs were already studied in [10, 11], but in the context of the generalization and systematicity. Following [10] we denote models with enhanced input representations by the "plus" sign.

## 2 Neural Prediction Machines

Techniques frequently used in RNN state space analysis are clusterization of the recurrent units activities and attributing meaningful information to the extracted clusters. Neural prediction machines [2] are models using clusters extracted from RNN state space as predictive contexts. The symbol just presented to the network corresponds to some cluster if and only if the RNN state belongs to this cluster. The next symbol probability of a symbol $a \in \mathcal{A}$ for a cluster $c$ is calculated by relating the number of times (counter $N_c^a$) when symbol $a$ follows a symbol driving the RNN to a given cluster $c$ with the number of times the RNN state belongs to the cluster $c$ ($\sum_{b \in \mathcal{A}} N_c^b$). The next symbol probability distributions are smoothed by applying Laplace correction parameter $\gamma$, hence probability of predicting symbol $a$ when in cluster $c$ can be calculated as:

$$P_{NPM}(a|c) = \frac{P(a,c)}{P(c)} \doteq \frac{\gamma + N_c^a}{\gamma A + \sum_{b \in \mathcal{A}} N_c^b}. \quad (1)$$

We set the Laplace correction parameter $\gamma$ to the value of $A^{-1}$. This can be seen as if we initialized counters to the value $\gamma$ prior to counting any symbol occurrences thus attributing some probabilities also to symbols not present in the training sequence. The more important the context, the less smoothing is performed. On the other hand, the probability distribution of rare (statistically less convincing) context is smoothed more heavily.

## 3 Echo State Networks

Echo state network is formed of one huge recurrent layer composed of hundreds or even thousands sparsely interconnected units. Input and recurrent connections initialized randomly with small weights and are not modified in the training phase. Only output weights are trained usually using simple linear regression. Simple and fast training is the most appealing feature of ESNs.

Experiments with ESNs were done in similar way as described in [12]. Output units had linear activation function, recursive least squares were used to train the network. When $\mathbf{u}(t)$ is an input vector at time step $t$, activations of internal units were updated according to

$$\mathbf{x}(t) = f_{\text{hid}}\left(\mathbf{W^{in}} \cdot \mathbf{u}(t) + \mathbf{W} \cdot \mathbf{x}(t-1) + \mathbf{W^{back}} \cdot \mathbf{y}(t-1)\right), \tag{2}$$

where $f$ is the internal unit's activation function, $\mathbf{W}$, $\mathbf{W^{in}}$ and $\mathbf{W^{back}}$ are hidden-hidden, input-hidden, and output-hidden connections' matrices, respectively. Activations of output units are calculated as

$$\mathbf{y}(t) = f_{\text{out}}\left(\mathbf{W^{out}} \cdot [\mathbf{u}(t), \mathbf{x}(t), \mathbf{y}(t-1)]\right), \tag{3}$$

where $\mathbf{W^{out}}$ is hidden-output and output-output connections' matrix.

The next symbol probabilities $p(t)$ in time step $t$ were calculated from activities on output units. First activities smaller than specified minimal value $o_{\min}$ were set to $o_{\min}$ and then probabilities were estimated by normalizing output acivities:

$$\hat{o}_i(t) = \begin{cases} o_{\min} \text{ if } o_i(t) < o_{\min} \\ o_i(t) \text{ otherwise} \end{cases}, \tag{4}$$

$$p(t) = \frac{\hat{o}_i(t)}{\sum_j \hat{o}_j(t)}, \tag{5}$$

where $o_i(t)$ is the activity of the output unit $i$ in time $t$. $o_{\min}$ was set to $0.001$ throughout all experiments.

## 4 Extracting Input Representations

An enhancement of ESN denoted as ESN+ was proposed in [10]. In this modification of ESN input weights were not initialized randomly but simple word co-occurrence statistics was used [13]. The value of each input weight $W^{in}_{i,j}$ was set to the ratio:

$$R_{i,j} = N \cdot \frac{N(i,j) + N(j,i)}{N(i) \cdot N(j)} \tag{6}$$

where $N(i,j)$ is the number of times symbol $i$ and $j$ occur next to each other and $N$ is the length of training sequence.

Since dimensions of matrices $\mathbf{R}$ and $\mathbf{W^{in}}$ are usually different remaining elements of $\mathbf{W^{in}}$ were set to zeros [10]. In our work we have slightly modified this approach and we initialized input weights as $\mathbf{W^{in}} = \mathbf{T^{rand}} \cdot \mathbf{R}$, where $\mathbf{T^{rand}}$ is transformation matrix created with small random values from $(-0.01, 0.01)$. Better results than with original method were achieved.

## 5 Datasets

The first dataset we used was the so called "Elman's Grammar" (EG) [14]. In [6] ESN's ability to learn grammatical structures was studied using this language. Dataset alphabet $A$ is composed 24 words including end-of-string marker. The training set was composed of 10000 sentences (55273 symbols) and the test set of independently generated 10000 sentences (54966 symbols). The entropy estimated on the test set is H = 0.534.

Language generated by complex stochastic context-free grammar (CG)[3] was used in experiments with models using self-organization [15]. Grammar was inspired by child-directed sentences with added recursive structures. The language vocabulary was composed of 72 words including the end-of-string marker. Similarly to [15] the training set was composed of 5000 randomly generated sequences (30985 symbols) and the test set comprises 10000 randomly generated sequences (62387 symbols). The estimated entropy is H = 0.398.

## 6 Experiments and Results

The predictive performance was evaluated using normalized negative log-likelihood (NNL). NNL was calculated as:

$$\mathrm{NNL} = -\frac{1}{T}\sum_{t=1}^{T}\log_{|A|} P_t(s_t) \approx -\frac{1}{T}\sum_{t=1}^{T}\sum_{a\in A} G_t(a)\log_{|A|} P_t(a), \tag{7}$$

where the base of the logarithm is the alphabet size $|A|$, $s_t$ is the symbol in the test set in time step $t$. $P_t(a)$ stands for the probability of predicting symbol $a$ in the time step $t$ and $G_t(a)$ is the ground true probability for symbol $a$ in time step $t$. NNL can be seen as the compression ratio, NNL of $0.5$ means that original sequence can be compressed to the half of its original size. Language entropy $H$ can be estimated as:

$$\mathrm{H} = \frac{1}{T}\sum_{t=1}^{T}\sum_{a\in A} G_t(a)\log_{|A|} G_t(a), \tag{8}$$

and can be seen as the best achievable NNL of a perfect model.

We tested ESNs with reservoirs of different sizes. Other parameters such as spectral radius, the sparsity of input and hidden weights, and input weigth range are of much less importance when processing symbolic sequences. For ESN we provide results for spectral radius of 0.98. The probability of existence of input weight was 1.0 and their values were choosen randomly (uniform distribution) from interval (-1.0, 1.0). We found out that the probability $p_{\mathrm{rec}}$ of existence of recurrent weight (determining sparsity of the recurrent weight matrix) has very little influence on ESN performance. To ensure full connectivity for small networks and a reasonable computation time for large networks we set $p_{\mathrm{rec}} = \min(1, 10/n)$, where $n$ is the number of reservoir units.

ESN+ seemed to be more sensitive to parameter settings. For EG dataset $p_{\mathrm{rec}}$ was increased to $p_{\mathrm{rec}} = \min(1, 50/n)$ and for CG dataset it was set to $1.0$. Also the spectral

[3] Generously provided by Igor Farkaš

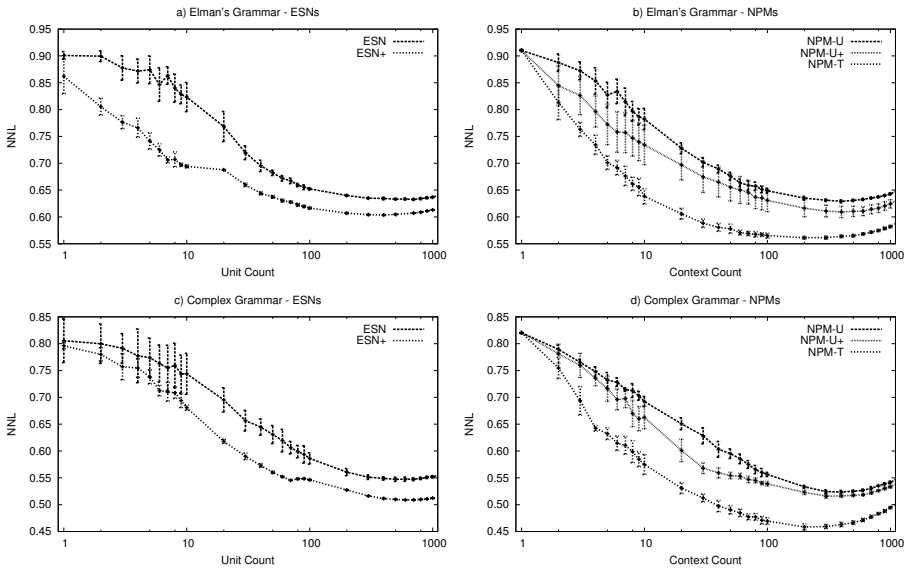


**Fig. 1.** Predictive performance of ESNs (left) and NPMs (right) on EG (top) and CG (bottom).

radius of ESN+ for CG dataset was changed to the value of $0.5$. Results are presented in Fig. 1a and Fig. 1c. ESN+ significantly outperforms ESN model.

NPM models with context count varying from 1 to 1000 were tested. NPM-U was built over untrained and NPM-T was built over trained recurrent part of SRN with 16 hidden units. The model with modified input representations is denoted as NPM-U+. We present mean and standard deviations of 10 simulations performed with the same parameters. NPM-T was build over SRNs carefully trained using Kalman filtration [16].

Not surprisingly NPM-Ts performed significantly better then NPM-Us. Careful adaptation process affected the RNN state space organization through adaptation of both input and recurrent weights and the trained SRNs (or models built over the trained SRNs) outperform other models. But improvement of NPM-U+ over NPM-U is also visible (mostly the case of EG dataset). Results of NPM models are presented in Fig. 1b and Fig. 1d.

## 7 Discussion and Conclusion

Results reveal an improvement of predictive performance of ESNs and NPM-Us can be achieved by creating proper input representations. For both EG and CG datasets ESN+ and NPM-U+ models performed significantly better than their counterparts consistently for all numbers of predictive contexts. Welch modification of unpaired Student's t-test was used to evaluate statistical significance and almost all comparisons proved to be significant at $p = 0.05$ and great majority of comparisons of models with context count above 10 proved to be significant at $p = 0.001$. In the following we attempt to analyze

state space representations of studied models to justify these preliminary results. We start by showing the impact of proper input representations by visualizing input weights initialized by different methods used in this paper (Fig. 2).

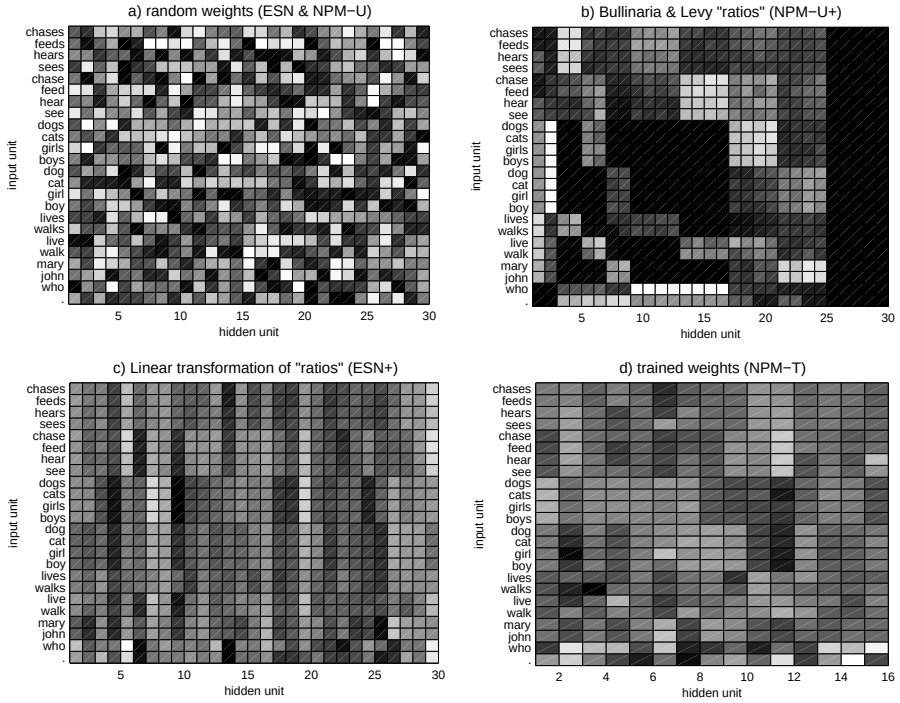


**Fig. 2.** Input representations of words from EG dataset created: a) randomly; b) Bullinaria & Levi "ratios" as used in [10]; c) linear transformation of "ratios" used in this work; d) by RNN training.

Standard ESN and NPM-U are built over untrained network, where input weights are chosen randomly (Fig. 2a). In the case of ESN+ and NPM-U+ co-occurrence statistics according to Bulinaria & Levy [13] is used (Fig. 2b). This results in similar input vectors for words from the same grammatical category (e.g. walks and lives, john and mary, boy and girl, etc.). In the case of ESN+ we apply linear transformation to scale down input weight values (Fig. 2c). Similar representation of input words can be also seen in NPM-T, i.e. NPM built over network trained by extended Kalman filter (Fig. 2d).

However, even when input representations in ESN+, NPM-U+ and NPM-T are identical, predictive performance of NPM-T is superior. To analyze impact of input representation on the network state space organization more precisely we locate and analyze fixed point in each model used in this paper. There were three typical configurations of fixed points as shown in Fig. 3.

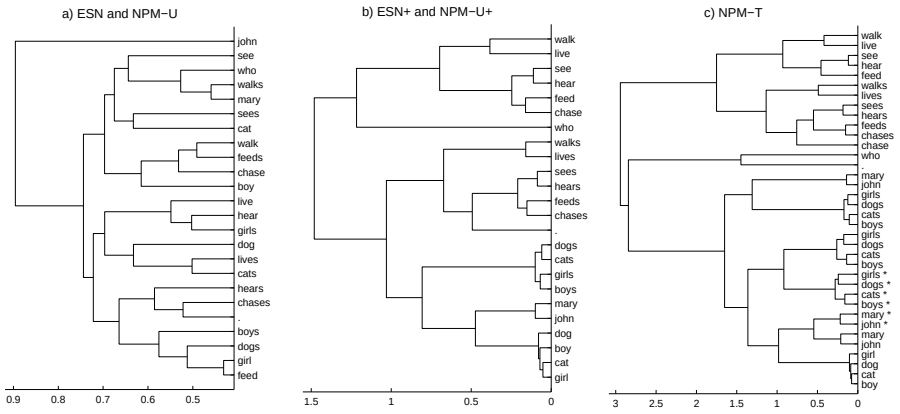


**Fig. 3.** Distance of fixed points in the network dynamics: a) ESN and NPM-U; b) ESN+ and NPM-U+ c) NPM-T. Several input words in NPM-T employ multiple fixed point in their dynamics (e.g. dynamics of words *john* and *mary* is composed of two attractors and one unstable saddle.)

The contractive dynamics of ESN and NPM-U models is simple. With fixed input, activities of hidden units converge to a single attractive fixed point in the network state space. Different input codes of words from $\mathcal{A}$ serve as bias which pushes attractive points in different directions. If input codes are initialized randomly, distances between attractors are random (Fig. 3a). When co-occurrence statistics is used, similar input vectors push attractive fixed points in the same direction. It means that the positions of attractors are still random, but now they are clustered according similar input words (Fig. 3b).

Finally, analysis of NPM-T revealed that the dynamics of underlying trained networks was not based on single attractive points. In Fig. 3c dynamics of several input words is composed of two attractors and one saddle combination (e.g. john, mary, girls, boys, etc.). Adaptation of both input and recurrent weights allows network to change contractive character of network dynamics and thus outperform other models.

In this work we build on results found in [9], where SRNs achieved better predictive performance by creating proper input representations during training process. Here we studied performance of models using contractive dynamics of untrained recurrent networks with inputs representations extracted from word co-occurrence statistics as described in [13]. Extraction method may not be optimal and more extensive study should be performed. Nevertheless models with extracted input representations perform significantly better then their counterparts with random input representations. Similar results but in the context of systematicity study were achieved in [10, 11].

**Acknowledgments.** This work was supported by the grants Vega 1/0848/08 and Vega 1/0822/08.

## References


1. Elman, J.L.: Finding structure in time. Cognitive Science **14**(2) (1990) 179–211
2. Tiňo, P., Čerňanský, M., Beňušková, Ľ.: Markovian architectural bias of recurrent neural networks. IEEE Transactions on Neural Networks **15**(1) (2004) 6–15
3. Ron, D., Singer, Y., Tishby, N.: The power of amnesia. Machine Learning **25** (1996) 117–149
4. Jaeger, H., Haas, H.: Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication. Science **304**(5667) (2004) 78–80
5. Jaeger, H.: Short term memory in echo state networks. Technical Report GMD 152, German National Research Center for Information Technology (2001)
6. Tong, M.H., Bickett, A.D., Christiansen, E.M., Cottrell, G.W.: Learning grammatical structure with Echo State Networks. Neural Networks **20** (2007) 424–432
7. Frank, S.L.: Strong systematicity in sentence processing by an Echo State Network. In Kollias, S., Stafylopatis, A., Duch, W., Oja, E., eds.: Artificial Neural Networks – ICANN 2006, Part I, Lecture Notes in Computer Science. Volume 4131. Berlin: Springer (2006) 505–514
8. Frank, S.L.: Learn more by training less: systematicity in sentence processing by recurrent networks. Connection Science **18** (2006) 287–302
9. Čerňanský, M., Makula, M., Beňušková, Ľ.: Organization of the state space of a simple recurrent neural network before and after training on recursive linguistic structures. Neural Networks **20** (2007) 236–244
10. S. L. Frank, M. Čerňanský: Generalization and systematicity in echo state networks. In: Proceedings of the 30th Cognitive Science Conference, Washington DC, USA. (2008) 733–738
11. S. L. Frank, H. Jacobsson: Sentence processing in echo state networks: a qualitative analysis by finite state machine extraction. Submitted to Journal of Algorithms (2008)
12. Jaeger, H.: The "echo state" approach to analysing and training recurrent neural networks. Technical Report GMD 148, German National Research Center for Information Technology (2001)
13. Bullinaria, J.A., Levy, J.P.: Extracting semantic representations from word co-occurrence statistics: a computational study. Behavior Research Methods **39** (2007) 510–526
14. Elman, J.: Distributed representations, simple recurrent networks, and grammatical structure. Machine Learning **7** (1991) 195–225
15. Farkaš, I., Crocker, M.: Recurrent networks and natural language: exploiting self-organization. In: Proceedings of the 28th Cognitive Science Conference, Vancouver, Canada. (2006) 1275–1280
16. Čerňanský, M., Beňušková, Ľ.: Simple recurrent network trained by RTRL and extended Kalman filter algorithms. Neural Network World **13**(3) (2003) 223–234